REMARKS

Petition for Extension of Time Under 37 CFR 1.136(a)

It is hereby requested that the term to respond to the Examiner's Action of October 28, 2008 be extended three months, from January 28, 2009 to April 28, 2009.

The Commissioner is hereby authorized to charge the extension fee and any additional fees associated with this communication to Deposit Account No. 50-4364.

Claims 22 and 23 were objected to as being of improper dependent form. These claims have been rewritten in independent form.

Claim 3 was objected to in view of its dependency upon claim 2, which referred to a "plurality" of compressed components. Claim 2 has been amended to refer to just a "component" that is decompressed, rendering the objection to claim 3 moot.

Claim 4 has been cancelled.

Claim 9 has been objected to on the basis that it describes components of the ROFS system forming part of the first data file (now the "first component"). The Examiner states that this is contrary to the description; however, it is explicitly stated at page 11, lines 26-28 of the "clean" version of the substitute Specification, that "certain executables and library files in the ROFS image may, preferably, also be shadowed entirely into RAM with the Core OS image" (emphasis added). There is therefore clear support in the description for the embodiment of claim 9.

Claims 2-24 have been objected to as failing to provide sufficient antecedent basis for the term "the further data file". Amended claim 24 refers to just one "further component"

(although other further components may, of course, also be present) in such a way as to provide proper basis for this term.

Claims 2-24 have further been objected to as failing to properly define the relationship between the "further data files" and the "further data file". The claims have now been amended for consistent and refer to just one "further component" – though of course more than one might actually be present.

Art Rejections

In the Office Action, the claims were rejected on the grounds that they were obvious in view of Maleug, or Maleug in combination with Chen under 35 U.S.C. §103. The Applicant respectfully disagrees with this assertion, and has amended the claims in order to clarify certain aspects that are novel and non-obvious over these documents. In particular, the claims now explicitly refer to first and further components of an operating system being accessed in different ways and yet still presented as part of a single file system (i.e. a "composite" file system). The effect of the different accesses has also been clarified – i.e. the distinction between 'permanent' and 'on demand' shadowing.

The present invention solves the objective technical problem of reducing the time required to load an operating system and boot up a computing device from non-executable memory, but without the complexity associated with the generation of the program code used by applications to be aware of which components have been loaded into executable memory, and which have not. A consequence of this is that there is more efficient use of the executable memory. The objective technical problem being addressed is not to increase the effective size of the execute-in-place RAM, such as provided by virtual memory techniques

for user data, or to provide a more flexible user selectable operating system, as described in the prior art.

The file server of a computing device allocates each storage medium on the device as one or more drives and the different media formats are implemented as separate file systems, in essence components which are plugged into the file server. Before a drive can be accessed, it must have a file system associated with it. When a drive has a file system associated with it, the drive is regarded as mounted in the device. The file server receives all client requests but does not access the media devices directly. The requests are passed to a file system. Each file system employs a media format which is appropriate for the characteristics of the devices that use it, and most are implemented as separate server plug-in DLLs, the exception usually being the ROM file system which usually is built as part of the file server itself.

Normally, a file server would mount two different file systems on two separate drives, but file server clients expect to find all the ROM components on a single drive. Hence, all of the ROM components, i.e. the components which are required to make up the operating system, are loaded onto a single drive in the RAM, which is the execute-in-place memory on the device. The drawback of this is that all of the OS components need to be loaded in one operation, with the attendant time delays before the device has booted up and become fully operational to a user.

A solution provided by the present invention is to shadow certain core OS components, referred to in the application as the Core OS image, into execute-in-place RAM on device start up, and to retain other components in the Non-executable memory, and load these subsequently on demand to supplement the Core OS components. These other

components are referred to in the application as the ROFS image. Each of these two sets of OS components has its own respective file system which usually would be presented to the file server as individual file systems, and would appear as separate drives. The file server provides access to a file system via a file/directory interface, which is very different to the mapping of virtual to physical address mapping of memory, as is used in paging and virtual memory systems.

However, the file server expects to see all of these components on a single drive with the file system associated with it, but this is not achievable when these components exist on separate drives on the execute-in-place and non-execute in place storage mediums. A solution provided by the present invention is to provide a composite file system which, in effect, is used to combine both file systems (the ROM file system and the ROFS file system) into a single file system, even though the operating system components addressed by those file systems exist on different drives.

In essence, the composite file system may be regarded as a thin processing layer which is used to pass requests from the file server to either or both of the ROM file system or the ROFS file system.

Maleug attempts to improve the efficiency of RAM usage by implementing a virtual memory system, where so-called 'XIP code' is demand-paged into RAM from a non-linear memory (such as NAND flash). In doing so, Maleug is able to provide a virtual RAM address-space that is far larger than the physical RAM. Maleug makes no teaching or suggestion of shadowing some code to the RAM permanently, whilst shadowing other code on an on-demand basis, as is claimed herein. Instead, Maleug describes code simply being

paged into the RAM as it is required. Indeed, at col. 5, lines 35-40, Maleug discusses the transient nature of the code stored in the RAM, stating that XIP code need not be properly paged-out of the RAM since it can simply be refreshed from its original location in the nonlinear memory when it is next needed (the code is immutable).

The Examiner has referred to col. 6, lines 13-31 of Maleug as disclosing the access of the "first data file" from the executable memory. This passage of Maleug refers to retrieving the boot loader from non-linear memory, and executing it in order to initiate the loading of the OS. Maleug does not, however, disclose permanently shadowing the boot loader into the RAM, and neither does Chen. Instead, the cited documents are silent as to whether the boot loader is copied into the RAM on a permanent basis, or whether it ceases to be resident in the RAM once its function is complete (the loading of the OS has been initiated).

Not only do Maleug and Chen both fail to teach or suggest that the boot loader is permanently shadowed in RAM, but Maleug also nowhere suggests that the boot loader is presented with the demand-paged XIP code as a single file system. Indeed, it would be incongruent for Maleug to so present the boot loader – it performs such a specialized and low-level task at such an early stage in the loading of the operating system that there would be nothing to be gained by providing such a composite file system. Indeed, amended claim 24 of the present application refers to copying the further component of the OS into executable memory "for subsequent use with the permanently shadowed first component [of the OS]". Since the sole purpose of the boot loader is to initiate the loading of the OS it makes no sense that it be "subsequently" used with components of the OS after it has loaded.

It will be understood, from the above, that the present invention (as clarified in the amended claims) addresses the problem of how to reduce the time taken to load an OS, and how to minimize the executable memory required by the OS, whilst also minimizing the complexity of the OS's programming. The present invention achieves this by dividing the OS into a permanently shadowed component that will always be resident in executable memory, and a component (or, in practice, many such components) that is shadowed only when it is needed, and then presenting all of these components as a single file system. Maleug also aims to reduce the executable memory required by a computer system, and does so by implementing a virtual memory technique between non-linear memory and RAM. However, since Maleug nowhere contemplates dividing an OS into components that are differently shadowed there is no need in Maleug to find a way to present such components as a single file system. Instead, Maleug can present a single file system simply by virtue of accessing all of the XIP-code using exactly the same demand-paging technique.

For the foregoing reasons, the Applicant respectfully submits that the claimed invention is novel and non-obvious over Maleug and Chen, taken alone or in combination.

Conclusion

The present invention is not taught or suggested by the prior art. Accordingly, the Examiner is respectfully requested to issue an early Notice of Allowance.

The Commissioner is hereby authorized to charge the extension fee and any additional fees associated with this communication to applicant's Deposit Account No. 50-4364.

Respectfully submitted

April 28, 2009
Date

/Mark D. Simpson/
Mark D. Simpson, Esquire
Registration No. 32,942

SAUL EWING LLP
Centre Square West
1500 Market Street, 38th Floor
Philadelphia, PA 19102-2189
Telephone: 215 972 7880
Facsimile: 215 972 4169
Email: MSimpson@saul.com